John von Neumann Institute for Computing

Massively Parallel Quantum Computer Simulations: Towards Realistic Systems

Marcus Richter, Guido Arnold, Binh Trieu, Thomas Lippert

published in

Parallel Computing: Architectures, Algorithms and Applications ,
C. Bischof, M. Bücker, P. Gibbon, G.R. Joubert, T. Lippert, B. Mohr, F. Peters (Eds.),
John von Neumann Institute for Computing, Jülich,
NIC Series, Vol. **38**, ISBN 978-3-9810843-4-4, pp. 61-68, 2007.
Reprinted in: Advances in Parallel Computing, Volume **15**,
ISSN 0927-5452, ISBN 978-1-58603-796-3 (IOS Press), 2008.

http://www.fz-juelich.de/nic-series/volume38

Massively Parallel Quantum Computer Simulations: Towards Realistic Systems

Marcus Richter, Guido Arnold, Binh Trieu, and Thomas Lippert

Jülich Supercomputing Centre (JSC)
Research Centre Jülich, D-52425 Jülich, Germany
E-mail: {m.richter, g.arnold, b.trieu, th.lippert}@fz-juelich.de

We discuss an extension of the *Massively Parallel Quantum Computer Simulator* by a gate level error model which covers operational errors and decoherence. Applying this error model to the Quantum Fourier Transformation (the kernel of Shor's algorithm) and Grover's quantum search algorithm, one finds that the QFT circuit is more robust to operational inaccuracies than Grover's algorithm on comparable scales. Critical parameters can be derived which give a first estimate of tolerable error thresholds. At present ion traps are regarded as the most promising technology for the realization of quantum computers due to the long coherence time of trapped ions. We discuss Hamiltonian based dynamical ion-trap simulations which have been developed in collaboration with the experimental working group of Prof. Rainer Blatt. In contrast to standard approaches no approximations like the rotating wave approximation or an expansion in the Lamb-Dicke parameter are required which allow for very accurate simulations. This permits to identify critical system parameters which limit the stability of the experiment.

1 Massively Parallel Quantum Computer Simulator

The *Massively parallel quantum computer simulator* is a software package available in Fortran 90 and C which has been developed to simulate universal quantum computers[3]. The software runs on various computer architectures, ranging from PCs to high-end (vector) parallel machines. The simulator can perform all the quantum operations that are necessary for universal quantum computation. The maximum number of qubits is set by the memory of the machine on which the code runs. On the supercomputer systems of the Research Centre Jülich `JUMP` and `JUBL` simulation results for quantum computers containing up to 37 qubits have been obtained. In view of the fact that the simulation of quantum systems, such as quantum computers, requires computational resources that grow exponentially with the system size, this represents a significant advance beyond the state of the art, which is currently around 32 qubits[4].

1.1 Quantum Computation

In a first step towards realistic quantum computer simulations we implement so called *ideal simulations*, where each gate is modeled by a quantum operation that acts instantaneously on the internal state of the quantum computer, neglecting both implementation imperfections and interactions with the environment.
In contrast to a classical bit the state of an elementary storage unit of a quantum computer, the quantum bit or qubit, is described by a two-dimensional vector of Euclidean length one. Denoting two orthogonal basis vectors of the two-dimensional vector space by $|0\rangle$ and $|1\rangle$, the state $|\psi\rangle$ of a **single qubit** can be written as a linear superposition of the basis states

$|0\rangle$ and $|1\rangle$:

$$|\psi\rangle_1 = a_0|0\rangle + a_1|1\rangle \ , \tag{1}$$

where the amplitudes a_0 and a_1 are complex numbers such that $|a_0|^2+|a_1|^2 = 1$. The state of a quantum computer with N qubits can be represented in the 2^N- dimensional Hilbert space as

$$|\psi\rangle_N = a_{0\ldots00}|0\ldots00\rangle + a_{0\ldots01}|0\ldots01\rangle + \ldots + a_{1\ldots10}|1\ldots10\rangle + a_{1\ldots11}|1\ldots11\rangle \tag{2}$$

Each operation on a quantum computer can be described by a 2^Nx 2^N dimensional unitary transformation $U = e^{-iHt}$ acting on the state vector $|\psi'\rangle = U|\psi\rangle$, with the hermitian matrix H being the Hamiltonian of the quantum computer model. An ideal quantum computer can be modeled by simple spin models such as the Ising model associating the two single-spin states $|\uparrow\rangle$ and $|\downarrow\rangle$ with the single-qubit basis states $|0\rangle$ and $|1\rangle$.

A quantum algorithm consists of a sequence of many elementary gates. These elementary gates are represented by very sparse unitary matrices. As the unitary transformation U may change all amplitudes simultaneously, a quantum computer is a massively parallel machine. A small set of elementary **one-qubit gates** (such as the Hadamard gate and the Phase shift gate) and a nontrivial **two-qubit gate** (such as the controlled NOT gate) are sufficient (but not unique) to construct a *universal* quantum computer. In the framework of ideal quantum operations any one- (two-) qubit operation can be decomposed into a sequence of 2x2 (4x4) matrix operations each acting on an orthogonal subspace of the 2^N dimensional Hilbert space.

1.2 Computational Resources and Performance

Due to the exponential growing Hilbert space with cumulative number of qubits, the simulation of quantum computers is clearly memory bounded. To represent a state of a quantum system of N qubits in a conventional, digital computer, we need at least 2^{N+4} bytes if each amplitude is represented by a complex double. Simple storage of the state vector in case of a 37 qubit system requires a memory of 2 TB. An efficient implementation of quantum operations on this state vector even requires 3 TB of memory (a detailed description of the implementation is given in[3]). In the following table we specify the typical simulation requirements depending on the system size N. The last row indicates the overall memory requirements to efficiently simulate quantum operations including the amount of memory to store the state vector.

#qubits N	**32**	**33**	**34**	**35**	**36**	***37***
#cpus (IBM p690+)	32	64	128	256	512	*1024*
#nodes (`JUMP`)[a]	1	2	4	8	16	*32*
#cpus (Blue Gene Light)	256	512	1024	2048	4096	*8192*
#nodes (`JUBL`)[b]	256	512	1024	2048	4096	*8192*
memory (state vector)	64 GB	128 GB	256 GB	512 GB	1 TB	*2 TB*
memory (operation)	96 GB	192 GB	384 GB	768 GB	1.5 TB	*3 TB*

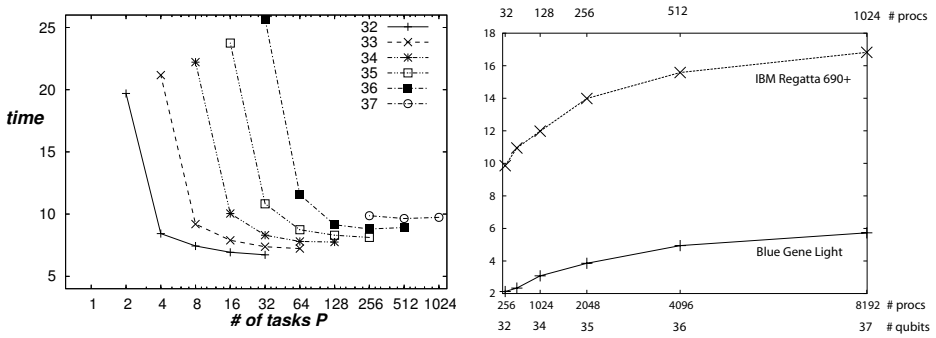

Figure 1. Left: average timings $t_{av}(N)$ on JUMP for a Hadamard operation on different system sizes N depending on the number of MPI tasks $P = 2^p$ using $T = 2^t$ OpenMP threads with $t + p = N - 27$. Right: scaling of the minimal average timings for the system sizes $N = 32, 33, 34, 35, 36, 37$.

From the user's perspective, the memory on a computer can be shared or distributed. On a shared memory computer, the state $|\psi\rangle$ of the quantum computer can be completely stored in the memory and all processors can access the entire memory. On a distributed memory machine, the elements of $|\psi\rangle$ are physically distributed over different nodes and each processor has direct access to its own local memory only. In the latter case, some extra programming is required to perform the communication between the processors. We use the standard Message Passing Interface (MPI) to perform the data communication.

Let us assume that the local memory (or more precise: one SMP-node) can store the 2^M amplitudes of the basis states of M qubits. Hence, to simulate a N-qubit quantum computer we need at least[c] $P = 2^N/2^M$ MPI processes. The amplitudes[d] $a_{x_{N-1}\ldots x_0}$ can be stored at the local memory address $A = \sum_{i=0}^{M-1} 2^i x_i$ of the MPI process with rank $R = \sum_{i=M}^{N-1} 2^{i-M} x_i$. In binary notation the local memory address and the rank of the processor reads $A = (x_{M-1} \ldots x_0)$ and $R = (x_{N-1} \ldots x_M)$, respectively. Recall that the qubits are numbered from 0 to $N-1$, that is qubit 0 corresponds to the least significant bit of the integer index, running from zero to 2^{N-1}, of the amplitude.

On a SMP-cluster like JUMP additionally OpenMP can be employed to parallelize the inter-node access. In this case the number of MPI-tasks obeys the obvious condition #nodes $\leq$ #tasks $\leq$ #CPUs. Since one JUMP-node can store the state vector of $N = 32$, a single CPU corresponds to $N = 27$ (maximal amount of memory per CPU) yielding

$$\underbrace{2^{N-32}}_{\#\text{nodes}} \leq \underbrace{2^{p}}_{\#\text{tasks}} \leq \underbrace{2^{N-27}}_{\#\text{CPUs}} \qquad \text{or} \qquad N - 32 \leq p \leq N - 27\,. \tag{3}$$

[a] The Juelich IBM p690+ (JUMP) is a cluster of 32 compute nodes each containing 32 Power 4+ processors (64bit) and 112 GB memory per node leading to 3.5 TB overall memory available to user access. Two processors share a L2 cache of 1.5 MB and each node shares a 512 MB L3 cache. Users normally only have access to max. 16 nodes equivalent to 512 processors.

[b] The Juelich Blue Gene Light System (JUBL) consists of 16384 PowerPC 440d CPUs driven by a clock rate of 700 MHz. Each node contains 2 processors which share 512 MB of memory. Depending on the application the nodes can be operated in two different modes: a coprocessor mode in which one CPU solely handles all the communication and a virtual node mode which allows to compute different tasks on each CPU.

[c] More processes are involved in case that the communication within one SMP-node is also realized via MPI.

[d] $x_i = 0, 1$ for $i = 0, \ldots, N-1$

The left half of Fig. 1 depicts the average timings for a single qubit quantum operation for different system sizes N against the number of MPI tasks $P = 2^p$. Almost in all cases the best performance is gained by choosing the maximal number of tasks. Only for $N = 36$ and $N = 37$ the performance increases marginally by using two OpenMP threads. The Juelich Blue Gene Light `JUBL` was operated in the coprocessor mode[e] in which one of the two CPUs per node handles all the communication. Therefore, OpenMP was not needed on this architecture.

The right side of Fig. 1 shows the scaling of the operation with increasing system size. Due to the little amount of memory per node, 8 times the number of CPUs is involved on `JUBL` compared to `JUMP`. Both architectures show nearly ideal weak scaling for a large number of qubits.

2 Operational Errors and Decoherence

Although the simulations described so far are gate-level based, it is nevertheless possible to include operational errors and decoherence effects. For this reason we have extended the *Massively parallel quantum computer simulator* by an error model which does not affect the intrinsic performance of the code.

To implement a basic model of operational errors every single qubit gate can be generated from *plane rotations*

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{4}$$

and *phase shifts*

$$P(\phi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}. \tag{5}$$

This decomposition allows to introduce Gaussian distributed angle- and phase errors ϵ with standard deviation σ, such that $R_\epsilon(\theta) = R(\theta + \epsilon)$ and $P_\epsilon(\phi) = P(\phi + \epsilon)$ respectively. Computation of controlled two- and more qubit gates can be reduced to effective single qubit gate computation acting only on that part of the state vector whose control-bit(s) are set to $|1\rangle$. For each quantum gate operation we draw ϵ_1 and ϵ_2 randomly from independent central Gauss distributions $\rho(\epsilon) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{\epsilon^2}{2\sigma^2})$.

A simple decoherence error model (depolarizing channel) allows for a *bit-flip* $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, a *phase-flip* $\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ or *both* $-i\sigma_y = \sigma_x\sigma_z = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ with probability $p/3$ each. The state vector remains unchanged with probability $1 - p$. It is replaced with a completely mixed state with probability p. In other words: The error randomizes the state with probability p. We assume an approximately constant operation time δt for every single gate independent of the type and the qubit it operates on. This duration fixes the time scale for our basic decoherence model. After each serial operation ($t = k\,\delta t, \quad k$=1,2,...) within the quantum circuit *each* of the n qubits (stochastically independent) can be subject to one of the depolarizing operators σ_α. For this we use n independent random sequences

[e] The performance was slightly better than in virtual node mode.

each containing m uniformly distributed numbers, where m is the size of the ensemble (=number of experiment repetitions).

In different experiments we study the effects of gate imperfections and decoherence depending on the standard deviation σ and the probability p. Given a certain confidence level we find out numerically thresholds for these parameters in real applications such as Quantum Fourier transformation or Grover's search algorithm. We compute the error-norm of the final state-vector as the average of m individual measurements $e^2(\sigma, p) = |\psi - \psi_{corr}|^2$.

These results are to be compared with future calculations from dynamic simulations of quantum computer devices, taking into account the full time evolution according to a time dependent Hamiltonian describing both, the system and the environment.

2.1 Quantum Fourier Transformation

The Quantum Fourier Transformation (QFT) is of particular interest since it represents the kernel of Shor's factorization algorithm[f]. To analyze the error robustness of the QFT circuit we plot the error-norm in dependence of (σ, p) for system sizes $n = 8, 16$ with 100000 repetitions and $n = 24$ with 10000 repetitions per experiment each starting from the initial state-vector $|000\ldots0\rangle$.

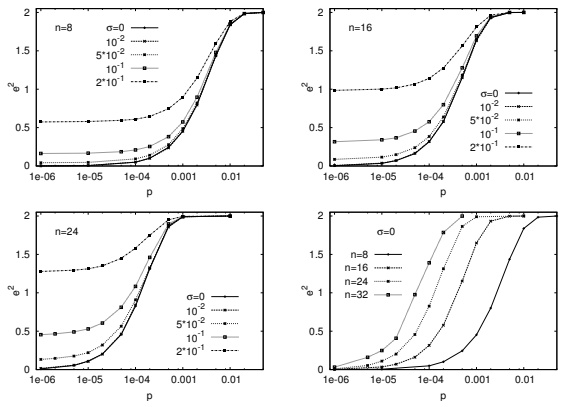

Figure 2. QFT: error-norm in dependence of σ and p for $n = 8, 16, 24$

Fig. 2 suggest a critical behaviour of the system in σ. The system is very robust against operational errors, since we find identical curves of $e^2(\sigma, p)$ for all $\sigma \leq 10^{-2}$ even for the largest system investigated. Larger operational errors increase the error-norm the more the larger the system is. To quantify the dependency of decoherence errors on the system size we have added the simulation results on a 32-qubit system gained from stochastic simulations of length $m = 1000$ each.

[f] Shor's algorithm allows to factorize natural numbers in polynomial time into a product of primes. This means, it could be used to break well-established encryption methods like the RSA scheme

2.2 Grover's Quantum Search Algorithm

Another well known quantum algorithm is Grover's algorithm for searching an unstructured database. In contrast to Shor's algorithm, the quantum search does not shift the problem into a different complexity class. Nevertheless, a quadratic speedup is gained which has been proven to be optimal.

Fig.(3) summarizes our simulation results for the system sizes 8+1, 16+1 and 23+1 qubits and demonstrates the effect of operational inaccuracies and decoherence errors on the amplitude $\psi(k)$ of the database element we are searching for. The (first) undisturbed maximum is expected after $l_{max} = 12, 201, 2274$ Grover iterations respectively. Due to the high cost for simulating larger systems we content ourselves with iterating the Grover circuit up to $l \approx l_{max}$ of the first period in case of n=16,23. The latter requires to simulate more than 35*10^6 quantum operations to collect statistics of m=100 runs. For the smaller systems we collect statistics of m=100000 (8+1) and m=10000 (16+1).

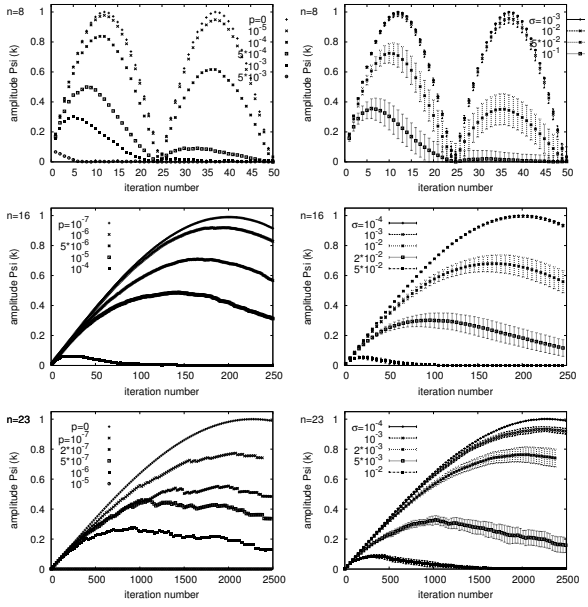

Figure 3. Decoherence (left column) or operational errors (right column) result in a damped amplitude $\psi(k)$ for the searched database entry k. The resultant maximum is shifted towards lower optimal number of Grover iteration steps.

In case of non-vanishing decoherence we can see damping of the value of the amplitude $|\psi_{max}(p, \sigma = 0)| < |\psi_{max,corr}| \approx \sqrt{1 - 1/N}$. The superposed decoherence process leads to a maximum shifted towards $l < l_{max}$ with a shift $\Delta = l_{max} - l$ growing for increasing system sizes n. We see amplitude damping in both cases, for operational and decoherence errors, but with very different sensitivities. In case of operational errors we state a more robust behaviour but switching to an appropriate deviation level we also see a clear shift of the maximal amplitude. To compare Grover's error sensitivity to the

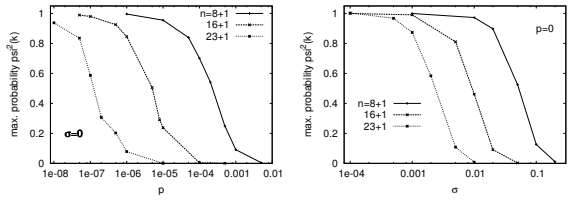

Figure 4. Comparison of the probability to find the correct database entry k for different system sizes in case of decoherence (left) or operational errors (right).

one of QFT we additionally plot the error-norm $e^2(\sigma, p) = |\psi - \psi_{corr}|^2$ in case of the 8+1 and (16+1) qubit system in fig.(5). In the presence of both error sources this investigation reveals a *system size dependent* threshold at $\sigma \approx 10^{-2}$ (and 10^{-3}) respectively. In contrast to the QFT algorithm this threshold is not constant but decreases significantly with the system size. For σ below these thresholds operational errors have nearly no impact. Beyond the threshold(s) we find a dramatically increased error-norm rising quicker with increasing system sizes than in case of QFT.

Asking for the maximal decoherence rate p at a given error tolerance e^2 in the under-critical σ regime the QFT algorithm allows a one (two) order(s) of magnitude higher decoherence probability respectively.

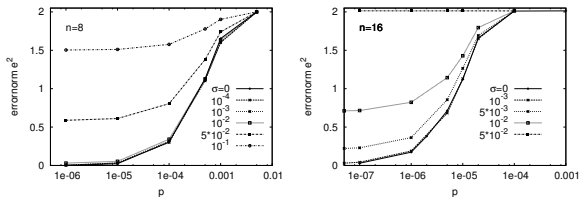

Figure 5. Comparing Grovers error-norm in the presence of both error sources for n=8,16 reveals a *system size dependent* threshold at $\sigma \approx 10^{-2}$ and 10^{-3} respectively.

3 Dynamic Simulation of Ion-Trap Devices

In order to simulate quantum devices from first principles, we have developed an additional package which efficiently solves the time-dependant Schrödinger equation. At present, ion traps are regarded as one of the most promising technologies for constructing a quantum computer, due to the long coherence time of the trapped ions. In a linear trap the ions (e.g. $^{40}\mathrm{Ca}^{+}$) are subject to a quasi harmonic potential $W(t)$ (see Eq.(7)) which keeps the ions fixed in space. The radio-frequency $\omega_{\mathrm{rf}}/2\pi$ induces only a so called micro-motion and is necessary, because in three dimensions it is impossible to trap electrically charged particles within a static potential. As long as the vacuum within the trap is sufficiently good, the ions are perfectly decoupled from the environment. Each ion represents one qubit in which the excitation levels $\mathrm{S}_{1/2}$ and $\mathrm{D}_{5/2}$ correspond to $|0\rangle$ and $|1\rangle$, respectively. The ions can be addressed via laser pulses (Eq. (9)) in order to induce internal transitions (cf.

Eq. (8)), excite bus phonons (which are needed to couple qubits for two-qubit operations), or measure a qubit.

The total time-dependant Hamiltonian consists of the following parts[7]

$$H_{\text{total}}(t) = H_{\text{motion}} + H_{\text{excitation}} + H_{\text{interaction}} \tag{6}$$

where

$$H_{\text{motion}} = \frac{p^2}{2m} + \frac{m}{2} W(t) x^2 \; ; \quad W(t) = \frac{\omega_{\text{rf}}^2}{4} \left[a_x + 2 q_x \cos(\omega_{\text{rf}} t) \right] \tag{7}$$

$$H_{\text{excitation}} = \frac{\hbar}{2} \tilde{\omega} \sigma_z \; ; \quad \tilde{\omega} = \omega_{\text{excited}} - \omega_{\text{ground}} \tag{8}$$

$$H_{\text{interaction}} = \hbar \Omega \sigma_x \cos(kx - \omega t + \phi) \; . \tag{9}$$

H_{motion} describes the collective motion of the ions in the trap potential $W(t)$, $H_{\text{excitation}}$ the internal excitation of the ions due to laser pulses and $H_{\text{interaction}}$ the laser ion interaction; σ_x and σ_z correspond to the Pauli matrices. Using a position space representation, the time evolution is calculated by a Lie-Trotter-Suzuki product formula (the momentum part $p^2/2m$ is solved via FFT).

In contrast to other approaches, no approximations – like the rotating-wave approx. or an expansion in the Lamb-Dicke parameter – are necessary, so that extremely accurate simulations can be carried out. First results demonstrate that effects which lead to a shift of the resonance frequencies – like the AC-Stark effect or off-resonant transitions – are correctly reproduced. This permits to identify critical system parameters which limit the stability of the experiment.

Acknowledgements

This research project was carried out in collaboration with the Computational Physics Group of Hans De Raedt (Univ. of Groningen) and Hartmut Häffner (Group of Rainer Blatt, Univ. of Innsbruck / IQOQI). We thank the DEISA consortium (co-funded by the EU, FP6 project 508830) for support within the DEISA Extreme Computing Initiative (`www.deisa.org`).

References

1. M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge (2000).
2. D. P. Di Vincenzo, `http://arxiv.org/abs/quant-ph/0002077`.
3. K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu, G. Arnold, M. Richter, Th. Lippert, H. Watanabe and N. Ito, *Massively Parallel Quantum Computer Simulator*, Comp. Phys. Comm. **176**, 121–136 (2007).
4. `http://www.qc.fraunhofer.de`
5. J. Niwa, K. Matsumoto, H. Imai, `http://arxiv.org/abs/quant-ph/0201042`.
6. G. Arnold, M. Richter, B. Trieu and Th. Lippert, *Improving Quantum Computer Simulations*, proceedings of the AQIS06 conference.
7. D. Leibfried, R. Blatt, C. Monroe and D. Wineland, Rev. Mod. Phys. **75**, 281 (2003).
8. J. I. Cirac and P. Zoller, Phys. Rev. Lett. **74**, 4091 (1995).